MS-I 2cy

[redacted]

N66-14299	
(ACCESSION NUMBER)	(THRU)
41	1
(PAGES)	(CODE)
CR 68972	10
(NASA CR OR TMX OR AD NUMBER)	(CATEGORY)

FACILITY FORM 602

GPO PRICE $ ________

CFSTI PRICE(S) $ ________

Hard copy (HC) 2.00

Microfiche (MF) .50

ff 653 July 65

[redacted]

GENERAL DYNAMICS

Convair Division

[redacted]

A2136-1 (REV. 5-65)

GENERAL DYNAMICS | CONVAIR

PHASE I FINAL REPORT

SIMULATION OF SELECTED DISCRETE NETWORKS

VOLUME ONE

THE LOGIC MODEL

CONTRACT NAS8-20016

Report No. GD/C DDF 65-005

Prepared by

GENERAL DYNAMICS/CONVAIR
A Division of General Dynamics Corporation
Huntsville, Alabama

for

George C. Marshall Space Flight Center
Huntsville, Alabama

September 1965

CONTENTS

ILLUSTRATIONS

TABLES

INTRODUCTION

The Discrete Network Simulation (DNS) system is based on simulation and analysis techniques developed for the Atlas Weapon System under government and corporate sponsorship. The total technique as applied to the Atlas Weapon System was called FASTI, Fast Access to System Technical Information. This study uses the Discrete Network Simulation portion with a modified version of the documentation and retrieval process. Digital computer programs are used to simulate discrete networks in less than real time. These programs were developed by GD/A and then incorporated into the FASTI system.

The prime purpose of Discrete Network Simulation methodology is to provide a set of analytical tools capable of conducting thorough, accurate and rapid analysis of complex systems. The methodology consists basically of:

A system network model.

A set of computer programs which will operate and activate the model.

These programs provide a realistic analysis and prediction of system performance before or after the hardware system is constructed. It is another form of testing; the results are as valid as those obtained by the more common hardware test procedures.

The Discrete Network Simulator (DNS) chronologically simulates events occurring due to the interactions among elements in a system network. Each "event," a Boolean change of state, is the result of a logical cause and effect relationship among elements in the system. The system modeled for the simulation may be a switching circuit, man/machine interaction, or any network where the component or subcomponent interrelations may be defined logically.

Convair is conducting a study under NASA Contract NAS8-20016, which applies the Discrete Network Simulation techniques to the Saturn SIC Engine Cutoff System networks. This report summarizes the results of Phase I and consists of three (3) volumes.

Volume One describes the methodology for constructing a DNS model.

Volume Two describes the DNS computer programs to the "Programmer." It is the "Users Reference Manual" for DNS.

Volume Three summarizes the study of the SIC Engine Cutoff System. The DNS Model and examples of the system simulation are described.

1/THE SYSTEM MODEL

The model is a series of Boolean algebra equations that logically describe the component interrelationships of a system network or partial system network. The Discrete Network Simulation programs (DNS) will chronologically simulate events occurring as a result of dynamic interactions among elements in a model. The use of the term, "Discrete Network," implies a system of variables defined in interdependent relationship. Each variable is discrete as its action is a binary event in the network: on or off, acting or not acting, available or not available, true or false, 1 or 0 etc. The variables represent events or activities. The network is described by: (1) a set of Boolean equations which completely define all interrelationships, and (2) a characteristic activity time associated with each variable (the time required for the effective change of state of a particular variable), which forms a complete mathematical model of a physical system.

1.1 DATA FOR MODEL BUILDING.

In order to develop a model, the following information must be available:

1.1.1 A complete set of circuit diagrams for the system to be modeled.

1.1.2 Information that shows mechanical to electrical ties and vice versa, such as limit and position switches for valves. If written description is not available, all the related mechanical drawings will be needed.

1.1.3 Design specifications for relays, automatic switches etc., where operating times are given.

1.1.4 An analysis prior to starting the model should be made to determine the boundary of the model, the number of variables that the model contains, and the inputs that will be necessary to make the model function.

1.1.5 The ground rules used to identify variables in the model can best be described by ref. to Figure 1-1, which is a portion of one of the Electrical Support Equipment drawings used to produce the model. The identifying names of the variables are divided, in general, into three parts as follows: (1) The engineering name by which

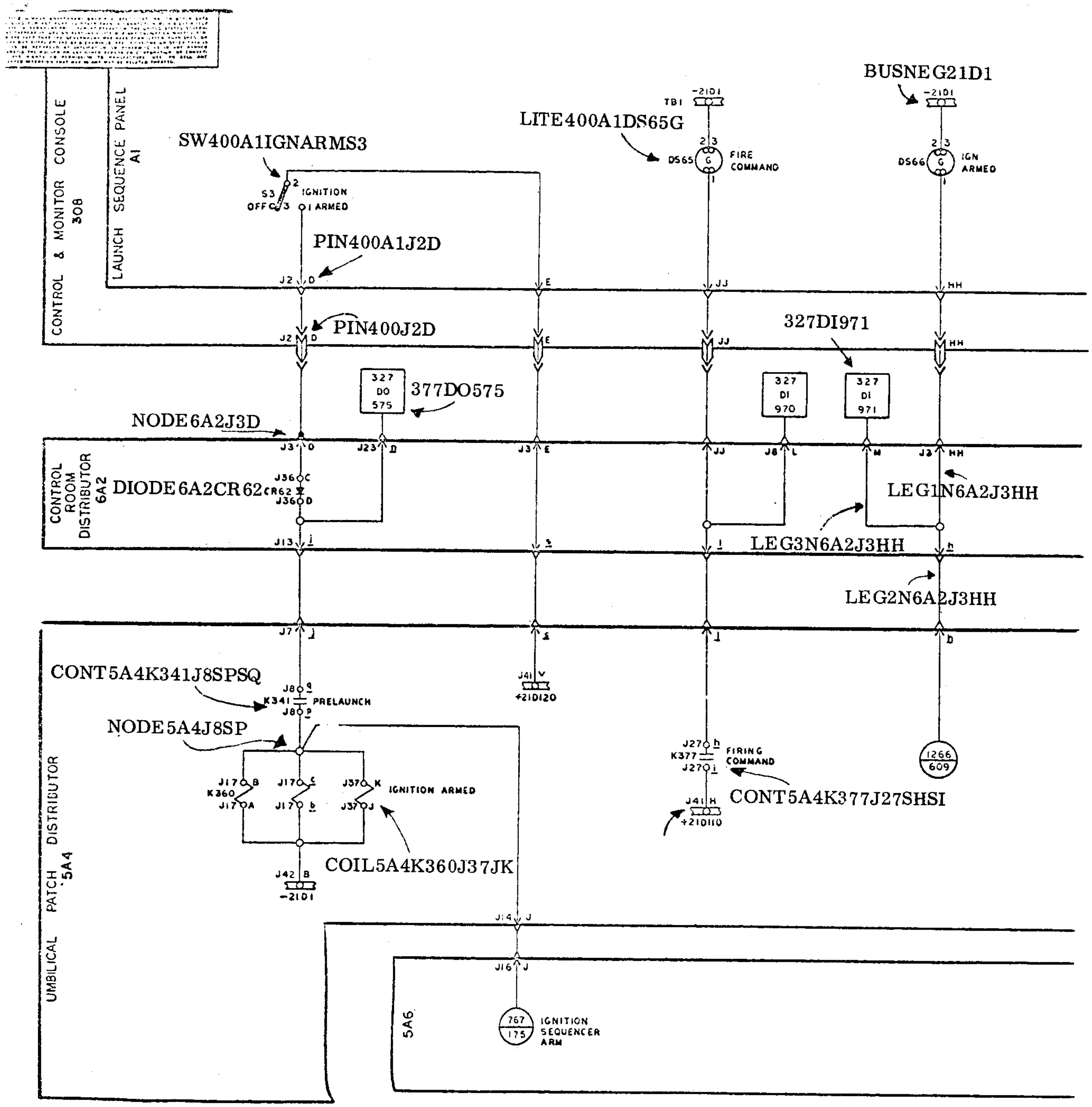

CONTROL & MONITOR CONSOLE 308
LAUNCH SEQUENCE PANEL A1
SW400A1IGNARMS3
S3
OFF
IGNITION ARMED
PIN400A1J2D
LITE400A1DS65G
DS65
FIRE COMMAND
BUSNEG21D1
DS66
IGN ARMED
TB1
-21D1
PIN400J2D
327DI971
377DO575
327 DO 575
327 DI 970
327 DI 971
NODE6A2J3D
CONTROL ROOM DISTRIBUTOR 6A2
DIODE6A2CR62
CR62
LEG1N6A2J3HH
LEG3N6A2J3HH
LEG2N6A2J3HH
CONT5A4K341J8SPSQ
K341
PRELAUNCH
+21D120
NODE5A4J8SP
K360
IGNITION ARMED
COIL5A4K360J37JK
-21D1
K377
FIRING COMMAND
CONT5A4K377J27SHSI
+21D110
1266 609
UMBILICAL PATCH DISTRIBUTOR 5A4
5A6
767 175
IGNITION SEQUENCER ARM

FIGURE 1-1

the variable would normally be called such as, COIL, DIODE, LITE, NODE, LEG, etc. Names such as CONTACT, SWITCH, etc., are abbreviated to keep the variable length below 24 characters. (2) The second part of the variable will normally be the designation of the hardware unit in which the variable is located with the designation as shown on the drawing, such as 5A7K402, 6A4CR3, 400A12DS89 etc., (3) The final part of the variable will pinpoint the exact location of the variable to a pin or terminal in a particular chassis.

The above rules apply to the majority of variables. In all cases they will be followed as closely as possible, to make the variable recognizable by anyone using the simulation. A variable must always be designated exactly the same in every equation or time-card in a model.

The following table shows how variables in Figure 1-1 were named:

ENGINEERING NAME	HARDWARE UNIT AND DRAW. DESIG.	UNIQUE LOCATION TO PIN OR TERM.
COIL	5A7K360	J17AB
COIL	5A7K360	* J17SBSC
COIL	5A7K360	J37JK
PIN	400A1	J2D
DIODE	6A2CR62	
LIGHT	400A1DS65G	
LIGHT	400A1DS66G	
NODE	6A2	J3D
NODE	6A2	J3JJ
CONT	5A7K377	J27SHSI
LEG1N	6A2	J3JJ
LEG2N	6A2	J3JJ
LEG3N	6A2	J3JJ
BUSNEG21D1		
BUS21D120		
327D1970		
327D0575		

TABLE 1-1

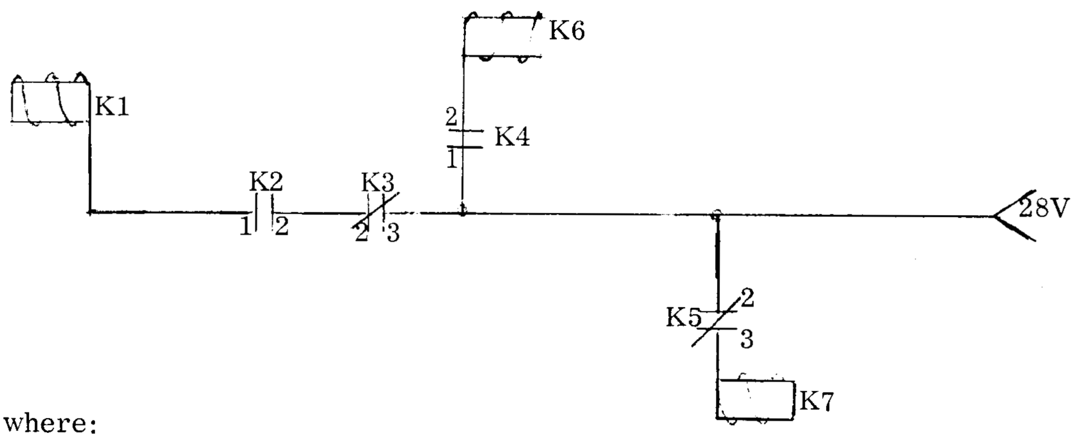
*Where lower case letters are used to designate a pin, an "S" is placed before the upper case of the same letter to better identify them in the model as the keypunch does not have lower case letters.

As can be seen from Table 1-1, all variables do not use every column of the allotted field. When buses are shown they have not been isolated to any particular chassis. Discretes were not tied to any chassis. The DNS Program does not allow a space between characters in a variable. There is no separation of the variables into three parts, as shown in the table.

2/BOOLEAN EQUATION AND TIME PARAMETERS

2.1 DEVELOPMENT OF BASIC BOOLEAN EQUATIONS USED IN A BASIC SIMULATION

Normally, all drawings show systems in a de-energized state. This is most evident with electrical drawings, for example:



where:

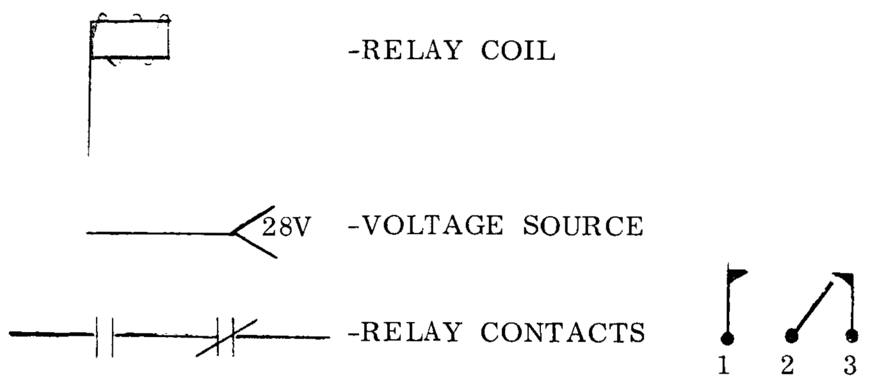


FIGURE 1-2

As shown, "1-2" contacts are open contacts when its relay coil is de-energized. Similarly, "2-3" contacts are closed contacts when its relay coil is de-energized. Therefore, to energize the above relay coil "K1," "K2" and "28V" must be energized, but not "K3." Stated again more discretely: K1 is energized when K2 is energized, K3 is not energized and 28V is energized.

2.2 SYMBOLS USED IN BOOLEAN EQUATIONS

The following basic symbols will be assigned for Boolean equations:

SYMBOL	MEANING
=	Equal
*	and
+	or
/	not

Restating the previous relationship: K1 is energized = K2 is energized * /K3 is energized * 28V is energized. The wording, "is energized," can be dropped, since it is common to all the variables in the equation, resulting in:

$$K1 = K2 * /K3 * 28V \tag{1}$$

which expresses all necessary actions required to energize "K1." This is the basic technique common to all Boolean equations for the S1C Engine Cutoff System. The equations are written describing the variable in its de-energized state. This is the state that is shown on most drawings.

2.3 BINARY REPRESENTATION OF VARIABLES

Because every variable in the system can only be described as "energized" or "not energized" (VARIABLE or /VARIABLE, respectively) and since all variables are described in Boolean Algebra, there are only two possible states for a variable. In Boolean Algebra the two states are represented by a "1" and a "0." "One" represents a signal, or energized, and "0" represents a lack of a signal, or not energized.

2.4 OPERATION OF THE SIMULATION

In the system simulation, a variable may change from a binary "1" to a binary "0" or from a binary "0" to a binary "1." The cause of this change could be from either the actions of associated variables or from actions of commands in the INPUT. The INPUT is a listing of commands which at the time specified by the particular INPUT will enter the simulation causing other variables to react. Should there be any contradiction caused by this entry, the input command will dominate.

Given the following circuit:

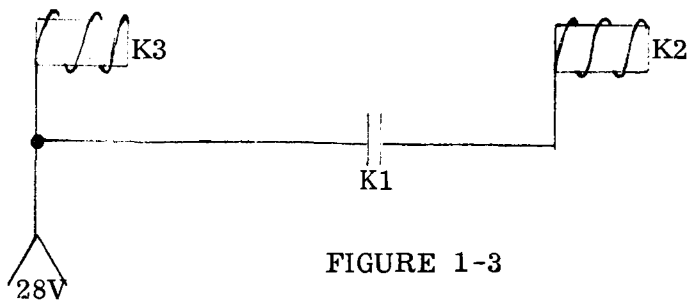


FIGURE 1-3

and the following INPUT commands:

28V = 1 at Time 12

28V = 0 at Time 259

the Boolean equations for "K2" and "K3" are:

$$K2 = K1 * 28V \tag{2}$$

$$K3 = 28V \tag{3}$$

In the system simulation, when the input command, "28V = 1 at Time 12," is applied to Eq. 2 and 3, action will be taken on Eq. 3, causing this variable to change from a binary "0" to a binary "1." There is no change to Eq. 2 since the variable K1 has not been energized. Should K1 later become a binary "1" before Time 259, Eq. 2 will then also become a binary "1."

At a later time prior to Time 259, should K1 become a binary "0," K2 will return to a binary "0." At time 259 when 28V becomes a binary "0," then K3 will become a binary "0."

2.5 TIME IN THE SIMULATION

In a system simulation there must be assigned to each variable a pickup and a dropout time, that is the time necessary for the variable to change its binary state. The pickup time is the time necessary for the variable to change its state from a binary 1 to 0, while the dropout time is the reverse. The pickup and dropout times are broken down further into minimum, average, and maximum, and when a simulation is made, any combination of these pickup and dropout times can be used. Therefore, if a relay, K1, has a nomimal pickup time of 20 ms with a tolerance of

$\pm$ 10% and a nominal dropout time of 10 ms but also with a tolerance of $\pm$ 10%, the six times associated with this variable would be:

Minimum Pickup Time			-- 18 ms
Average	"	"	-- 20 ms
Maximum	"	"	-- 22 ms
Minimum Dropout Time			-- 9 ms
Average	"	"	--10 ms
Maximum	"	"	--11 ms

Having chosen the average time category and having written an equation describing the previous relay K1 (Figure 1-3), the pickup time is 20 ms. The Boolean equation stated for K1 was:

$$K1 = K2 * /K3 * 28V \tag{1}$$

When K2 and 28V are energized and K3 is not energized, K1 tries to become energized. This simulates the completion of the "copperpath" to the coil of relay K1. However, due to the mechanical delay of the relay, its contacts do not change their position for 20 ms. Therefore, in the simulation K1 is delayed from being energized (represented by a "1") for 20 ms. Similarly, should the copper-path open up as a result of a change in any of the variables, K2, K3 or 28V, the relay K1, will try to de-energize. Mechanical action will delay the contacts from changing by 10 ms. Therefore, in the simulation K1 will be delayed from being de-energized (represented by a "0") for 10 ms.

3/WRITING EQUATIONS

The methods used to write this model from the basic concept of the elementary Boolean equations are explained step by step in the following paragraphs.

3.1 ALL VARIABLES ARE DESCRIBED

Given the following section of a circuit, it is described as follows:

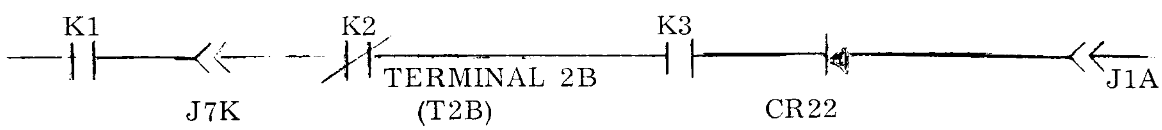


FIGURE 1-4

K1 * /J7K * /K2 * /T2B * K3 * /CR22 * /J1A

Similar to the previous representation of contacts and switches, the "logical negation " (slash symbol) of the extra variables only indicates that those variables are normally closed in their de-energized state (as shown on the drawing). In addition, if these types of variables are sensed in the simulation, because they are "logically negated" no additional logic or commands are necessary for normal or correct functioning of the system. However, the including of all variables in the original writing of the Boolean equations provides three additional benefits:

1. More valid simulation and malfunction analysis of the system.
2. More complete picture of the circuit for automatic display purposes for use in a display system such as the SC-4020.
3. More consistency provided by insuring that all model builders describing the same system include in the model all variables instead of certain specified types.

3.2 ALL VARIABLES ARE WRITTEN IN SEQUENCE

Consider again the previous circuit:

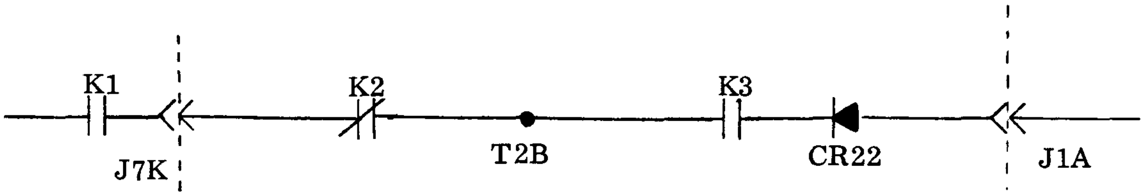


FIGURE 1-5

The Boolean equation for this circuit was:

K1 * /J7K * /K2 * /T2B * K3 * /CR22 * /J1A

Now notice that in going from left to right, the sequence of variables in the equation follows exactly the position of the variables in the circuit.

3.2.1 As will be discussed later in this report, this equation will be written in the opposite direction, such as:

/J1A * /CR22 * K3 * /T2B * /K2 * /J7K * K1

Regardless of which direction the equation is written, it is to be noted that the variables consistently follow each other sequentially as shown in the circuit drawing.

3.2.2 The sequencing of variables within the equation has the following advantages:

1. As a consistency rule, all equations will be written in the same order.
2. If the system is to be pictorially displayed, the display will correspond exactly to the sequencing shown on the circuit drawing.

3.3 LEG-NODE RELATIONS

Given the following circuit:

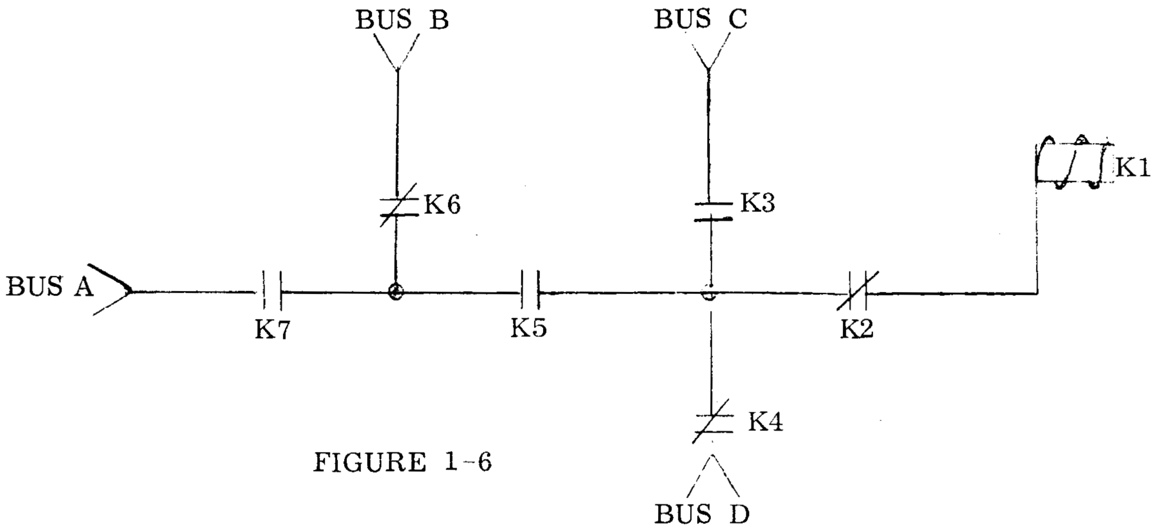


FIGURE 1-6

The basic Boolean equation used to describe this circuit is:

$$K1 = /K2 * (K3 * Bus\ C + /K4 * Bus\ D + K5 * (/K6 * Bus\ B + K7 * Bus\ A)\ ) \quad (4)$$

All equations are written from the variable being described to the first source of energy such as a "bus." In the above case, since there were four parallel paths, there were four "first" sources of energy, or four buses. This type of equation writing cannot only be cumbersome because of possible lengthy equations with many parallel paths, but also creates redundant writing. For example, if another coil, say Kx, were connected in parallel with Kl, the equation for Kx would be an exact copy of the equation for Kl, a complete redundancy.

3.3.1 The method used to describe this circuit introduces two additional type variables used throughout the model, namely:

1. The LEG

2. The NODE

They are defined as follows:

1. The LEG -- a single serial path between two nodes, a node and an initiator (source), a node and a terminal (lamp), or an initiator and a terminal.

2. The NODE -- the intersection of three or more legs.

3.3.2 Adding a little more labeling to the last diagram generates the following:

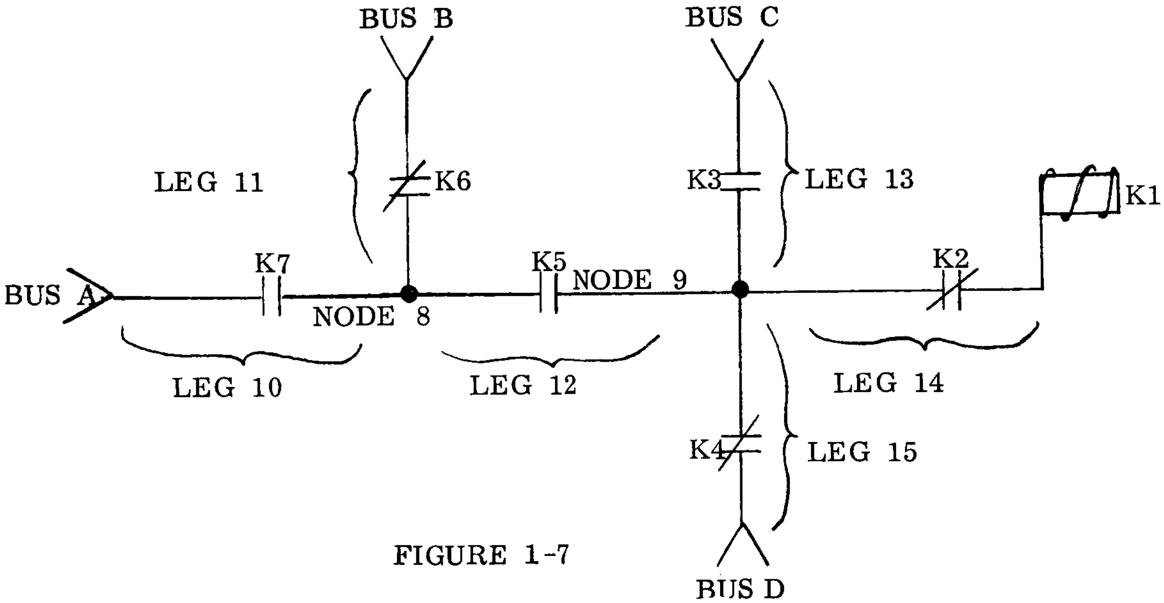


FIGURE 1-7

Figure 1-7 shows that there are only two nodes and six legs. Node 8 has three legs -- 10, 11, and 12 -- while Node 9 has four legs -- 13, 14, 15, and the common leg, 12.

3.3.3 The new equations describing this circuit now are:

Node 8 = Leg 10 + Leg 11 + Common Leg 12 (5)

Node 9 = Leg 13 + Leg 14 $\pm$ Leg 15 + Common Leg 12 (6)

Leg 10 = K7 * Bus A (7)

Leg 11 = /K6 * Bus B (8)

Leg 13 = K3 * Bus C (9)

Leg 14 = /K2 * K1 (10)

Leg 15 = /K4 * Bus D (11)

$$\text{Common Leg 12} = \begin{array}{c}\text{K5 * Node 8} \\ \text{or} \\ \text{K5 * Node 9}\end{array} \qquad (12)$$

The next section will show how the common Leg 12 is handled. The leg-node concept is similar to the type of equations developed by Kirchoff for circuit analysis. It allows systems or circuits to be described as two-way paths (bilateral).

3.4 BILATERAL TECHNIQUES

Given the following circuit:

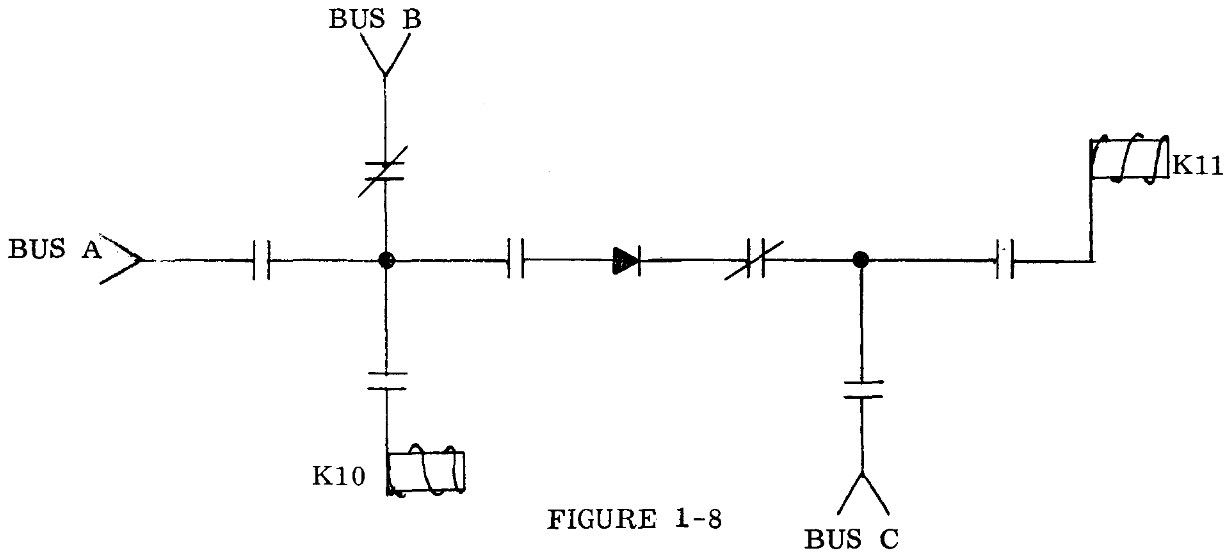


FIGURE 1-8

In the basic method of writing Boolean equations, only an energized electrical path (copper-path) from Bus A or Bus B could energize Coil K 10. This was due to the fact that all equations were written in a "forward manner," i.e., "normal flow of conventional current," or "from higher potential to a lower potential direction," or in a manner expressing only the "normal functioning of the system." Thus the Boolean equations could not show that Coil K10 could be energized by a copper-path from Bus C. But now suppose the diode shorts and a malfunction analysis is required. There is then a need to write many of the equations not only in a "forward-manner" but also in a "reverse manner" to show these "sneak-circuits."

3.4.1 Now suppose the circuit looked like this:

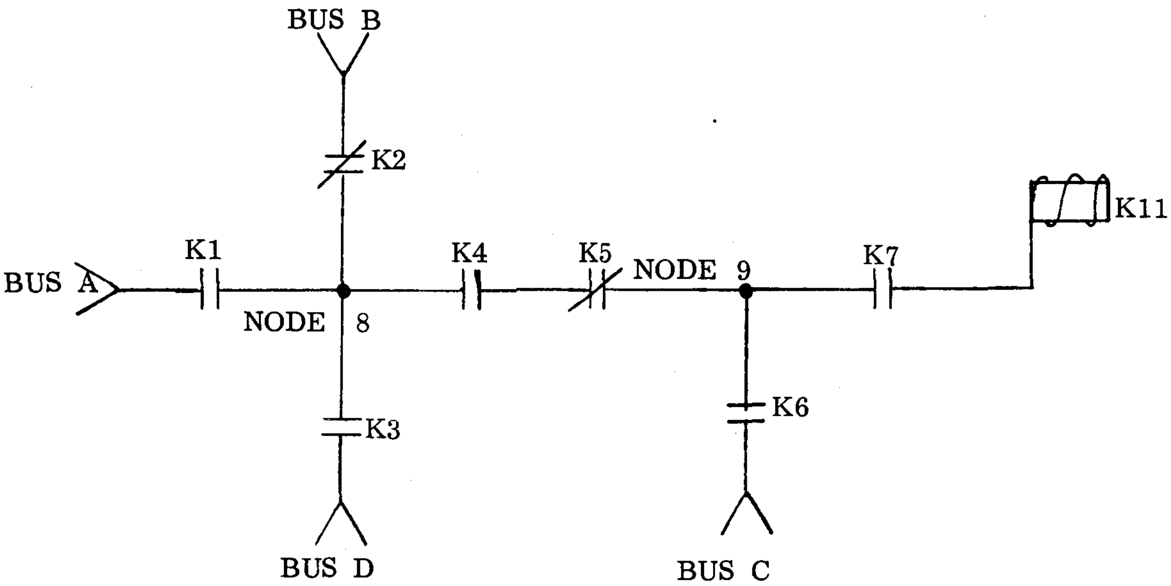


FIGURE 1-9

Even with the diode out of the circuit, with the previous equations, Node 8 could not be energized from the direction of Node 9. But these equations can be written bilaterally, that is, from two sides. From the back to the front (Forward) and from the front to the back (Backward). These equations might then look like:

(Forward): Node 9 = /K5 * K4 * Node 8 (13)

(Backward): Node 8 = K4 * /K5 * Node 9 (14)

But from the previous section it has been shown that:

1. Legs depend upon variables.

2. Nodes depend upon legs.

So these forward and backward equations are really forward and backward legs. They can be expressed as:

Leg of NODE 9 from NODE 8 = /K5 * K4 * NODE 8

Leg of NODE 8 from NODE 9 = K4 * /K5 * NODE 9

3.4.2 To avoid the apparent effect of current flowing in two directions simultaneously and, in addition, the inevitable "hold-in" feature inherent with this set of equations, they should be written as:

Leg (NODE 9 - NODE 8) = /K5 * K4 * NODE 8*/Leg (NODE 8 - NODE 9)

Leg (NODE 8 - NODE 9) = K4 * /K5 * NODE 9*/Leg (NODE 9 - NODE 8)

The node equations should be placed before the leg equations in the model so that the Boolean equations will give the right indications during the simulation when the system is being de-energized. Not all legs have to be described in a bilateral manner. Two types of legs that can be written one way only are:

1. Initiating Legs

   These are legs which are normally energized by an initial variable such as a bus or a source for which there is no further logic to describe this initial variable.

2. Ground Legs

   These are legs which are composed of variables from the grounded side of the hardware. This type of circuitry is further explained in a later section of this report.

3.4.3 It can be seen that legs not only have a name but also a direction associated with them. To minimize the name length of this type of variable it can be decided to:

1. Number all legs around all nodes independently and sequentially starting at "1" for each node at "12 o'clock" and going clockwise.

2. Within the name of these legs, replace the first half or the half telling the direction of the leg by this numbered leg.

For example:

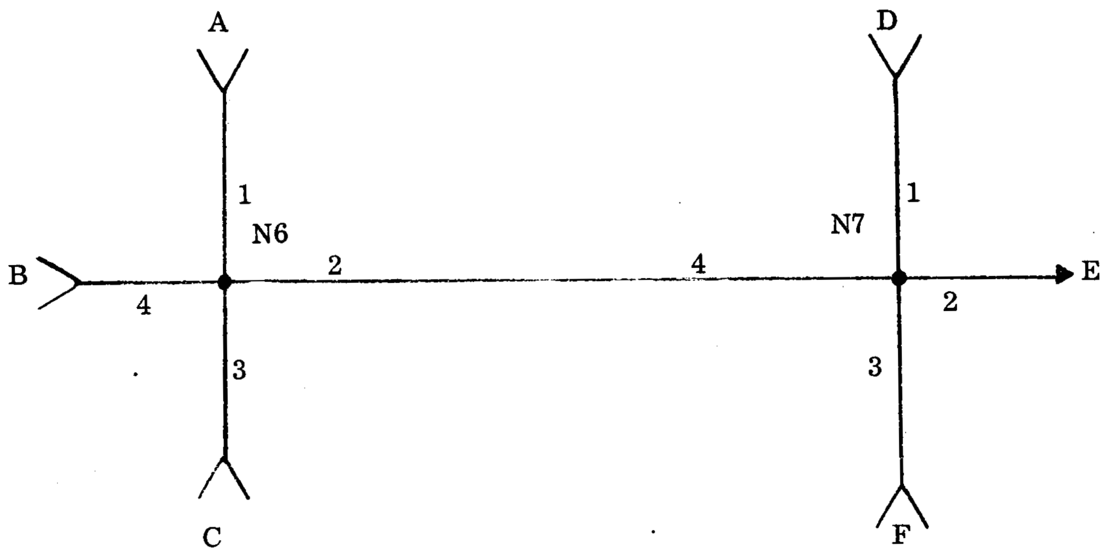


FIGURE 1-10

(Note: There are two numbers for common legs between two nodes.)

The designation of the leg equations of the two nodes shown in Figure 1-10 is as follows:

$$\text{Leg (N6-A} = \text{LEG1N6} \tag{15}$$

$$\text{Leg (N6-N7)} = \text{LEG2N6} \tag{16}$$

$$\text{Leg (N6-C)} = \text{LEG3N6} \tag{17}$$

$$\text{Leg (N6-B)} = \text{LEG4N6} \tag{18}$$

$$\text{Leg (N7-D)} = \text{LEG1N7} \tag{19}$$

$$\text{Leg (N7-E)} = \text{LEG2N7} \tag{20}$$

$$\text{Leg (N7-F)} = \text{LEG3N7} \tag{21}$$

$$\text{Leg (N7-N6)} = \text{LEG4N7} \tag{22}$$

3.4.4 Consider again one of the previous circuits:

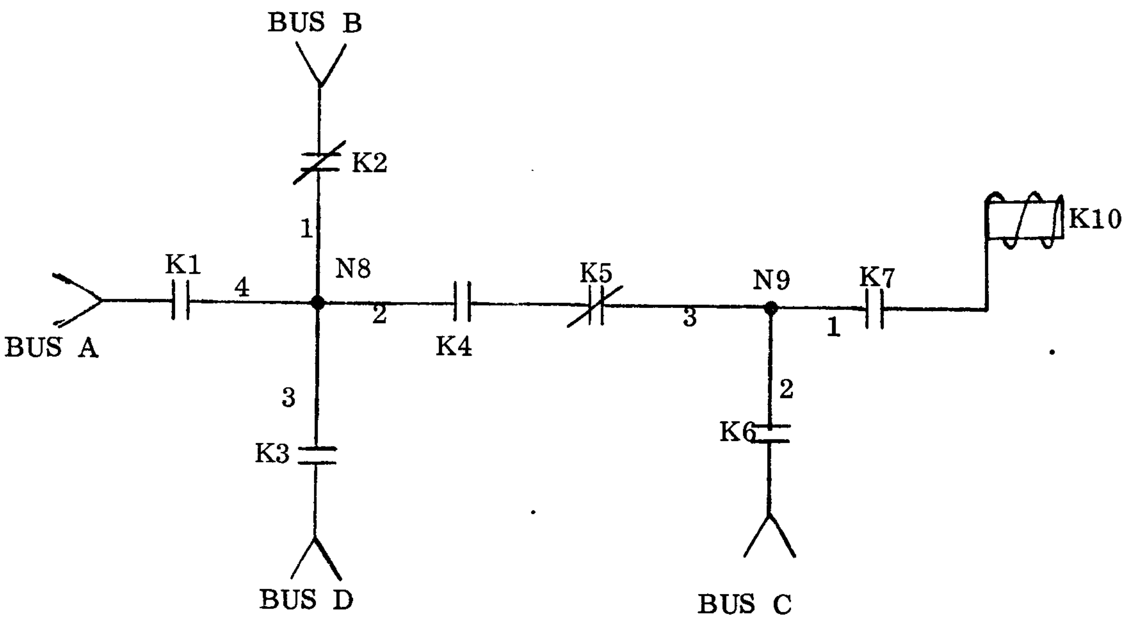


FIGURE 1-11

In summary, the node equations of this circuit are:

$$N8 = LEG1N8 + LEG2N8 + LEG3N8 + LEG4N8 \tag{23}$$

$$N9 = LEG1N9 + LEG2N9 + LEG3N9 \tag{24}$$

The initiating legs are:

$$LEG1N8 = /K2 * Bus\ B \tag{25}$$

$$LEG3N8 = K3 * Bus\ D \tag{26}$$

$$LEG4N8 = K1 * Bus\ A \tag{27}$$

$$LEG2N9 = K6 * Bus\ C \tag{28}$$

The sets of bilateral legs are:

$$LEG3N9 = /K5 * K4 * N8 */LEG2N8 \tag{29}$$

$$LEG2N8 = K4 * /K5 * N9 * /LEG3N9 \tag{30}$$

and

$$LEG\ (K10\text{-}N9) = K7 * N9 \tag{31}$$

$$LEGIN9 = K7 * K10 \tag{32}$$

The last set of bilateral legs will be further discussed in a later section, since it constitutes a deviation from a simple set of bilateral equations. Notice that not only the variables in the legs are in strict sequence as defined before, but the legs making up the node equations are in a clockwise numerical sequence.

3.5 DIODE ANALYSIS

3.5.1 Previously, diodes were taken into account when writing Boolean equations but they were never considered as a circuit element to be described. Diodes can actually be in any one of three states:

1. Normal

2. Open

3. Shorted

The following circuit can be used as an example:

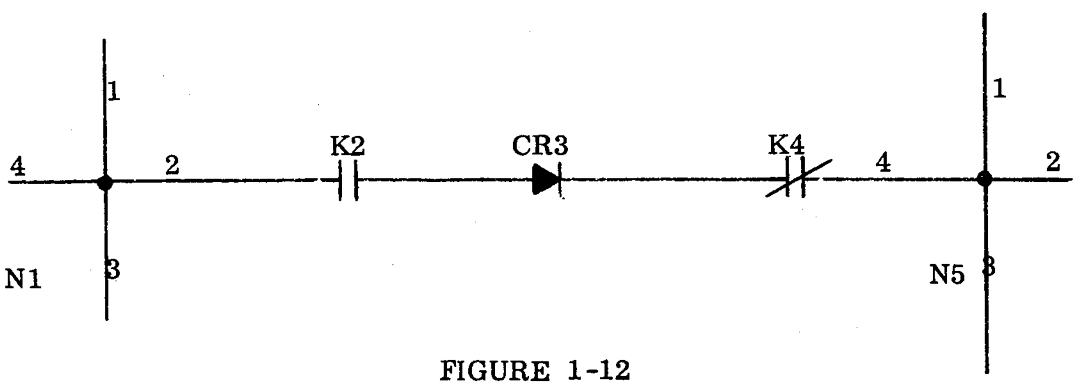


FIGURE 1-12

3.5.2 Since for a diode there are three states (as listed above) instead of two, at least two variables must be used to describe these three states. By not creating a name for the diode in the normal state and writing the equations for this leg in the following manner, a very efficient set of equations evolves to show the three states of the diode.

LEG4N5 = /K4 * /Diode CR3 Open * K2 * Nl * /LEG2Nl (33)

LEG2Nl = K2 * Diode CR3 Shorted * /K4 * N5 * /LEG4N5 (34)

The results can be tabulated thus:

1. Diode Normal -- no diode inputs needed for simulation. (Only forward leg allowed to energize.)

2. Diode Open -- Diode CR3 Open = 1. (Neither leg allowed to energize.)

3. Diode Shorted -- Diode CR3 Shorted = 1. (Only first leg to energize allowed to energize.)

3.6 COIL-CONTACT RELATIONS

Given the following circuits:

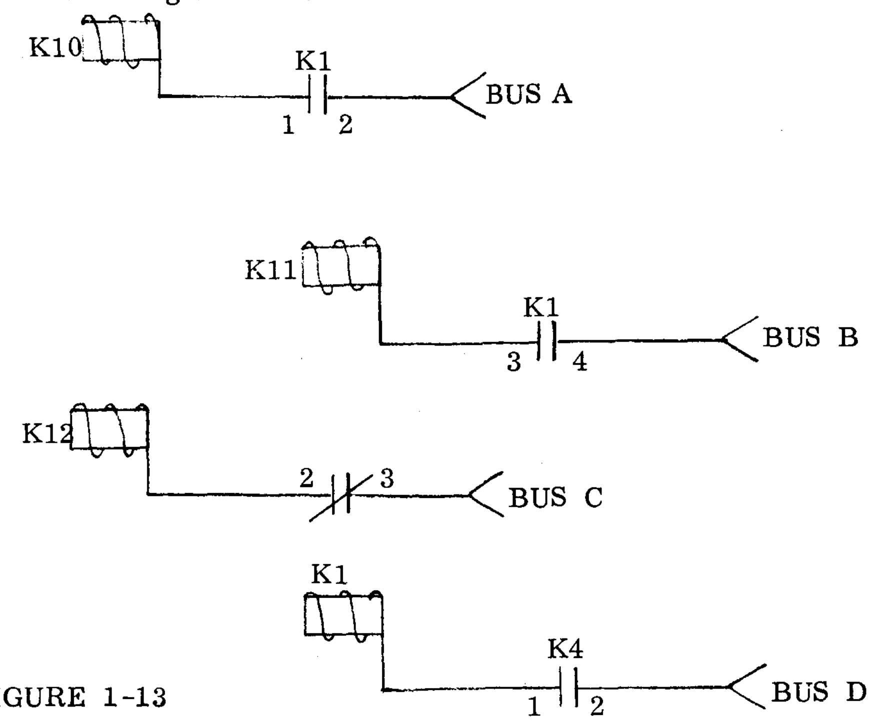


FIGURE 1-13

3.6.1 The coil equations will be written in the following manner:

COILK10 = CONTK1C12*BusA (35)

COILK11 = CONTK1C34*BusB (36)

COILK12 = 1CONTK1C23*BusC (37)

COILK1 = CONTK4C12*BusD (38)

3.6.2 In addition, the contacts have to be related logically to the coil. This is accomplished by equating each set of contacts of a particular relay to the relay.

CONTK1C12 = COILK1 (39)

CONTK1C34 = COILK1 (40)

CONTK1C23 = COILK1 (41)

3.6.3 When writing those Boolean equations which include contacts as shown on the drawing, it is a good idea to check those contacts with the relay table, showing every set of contacts for each relay and whether the contact set is supposed to be open or closed in the de-energized state. Most drawings will show the contacts and other elements and circuits in the de-energized state.

When writing legs in a bilateral fashion and encountering the contacts of a coil, be sure that the contacts are written in the same order. For example:

FIGURE 1-14

If the forward equation is:

* /CONTK2C67 * CONTK1C34*

then the backward equation should be:

* CONTK1C34 * /CONTK2C67*

The pickup and dropout times are placed on all sets of contacts for the coils, while no pickup nor dropout time is placed on the coil.

3.7 GROUND CIRCUITS

Ground circuits should be described as follows:

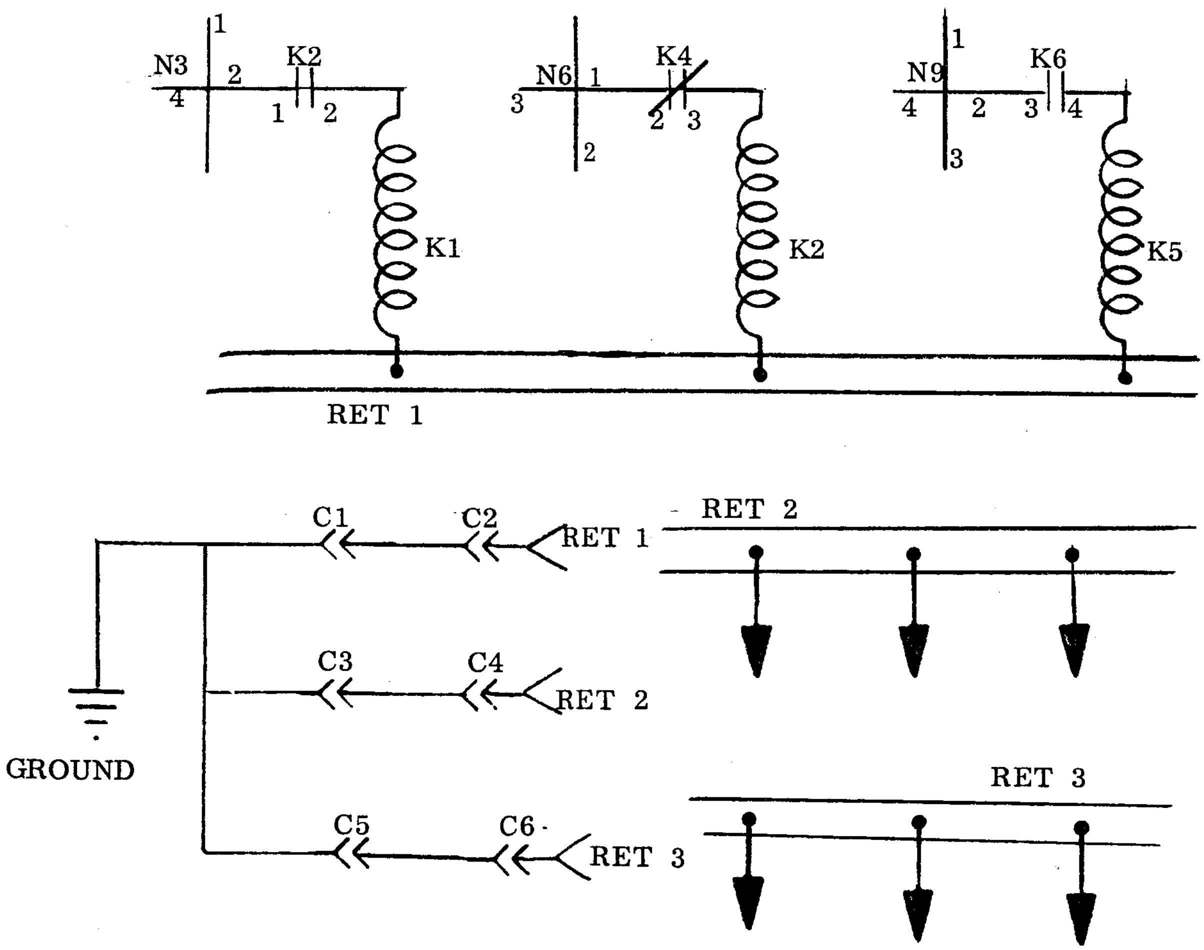


FIGURE 1-15

3.7.1 Previously, ground circuitry has been ignored. It was assumed static, always there when needed. But this is not always true. Sometimes the whole circuit can not be energized until a switch is closed in the grounded side of the circuitry. Also, from a malfunction analysis point of view, there are some types of malfunction that can occur in ground circuitry, and so ground circuitry should be expressed. The forward and backward Boolean equations for a variable, such as a coil, should be written as follows:

Leg (K1-N3) = CONTK2C12 *N3 (42)

LEG2N3 = CONTK2C12 * Coil K1 (43)

But now this is only saying that a variable can be energized when there is power applied, ignoring the return circuit. To make the model more complete:

Leg (K1-N3) = Ret 1 * CONTK2C12 * N3 (44)

LEG2N3 = CONTK2C12 * Coil K1 * Ret 1 (45)

Notice that in the forward leg equation, the ground circuit variable, Ret 1, appears first and because of the reverse sequence, appears last in the backward leg equation. Similarly, for the other variables requiring a return circuit, the equations are:

Leg (K2-N6) = Ret 1 * /CONTK4C23 * N6 (46)

LEG1N6 = /CONTK4C23 * Coil K2 * Ret 1 (47)

Leg (K5-N9) = Ret 1 * CONTK6C34 * N9 (48)

LEG2N9 = CONTK6C34 * Coil K5 * Ret 1 (49)

In this form these bilateral equation sets are subject to the "hold-in" feature.

3.7.2 The rest of the ground circuitry need only be described in a unilateral manner eliminating node and leg designations:

Ret 1 = /C2 * /C1 * Ground (50)

Ret 2 = /C4 * /C3 * Ground (51)

Ret 3 = /C6 * /C5 * Ground (52)

The variable "Ground" is an input or initiator, and there is no logical equation describing it, just as there are no logical equations describing the mechanical connectors, Cl, C2, ·C3, C4, C5, and C6.

3.7.3 In addition, there are other considerations for equations involving coils, lamps, motors, and other "power absorbing" variables. This type of variable is discussed more fully in the following section and the final form of equation will be written, including a ground circuit variable.

3.8 "SINK" EQUATIONS

3.8.1 "Sink" equations include bilateral leg equations for all power absorbing variables, such as:

1. Coils
2. Lamps
3. Motors
4. Discrete Inputs
5. Others

3.8.2 A "sink" leg is shown in Figure 1-16.

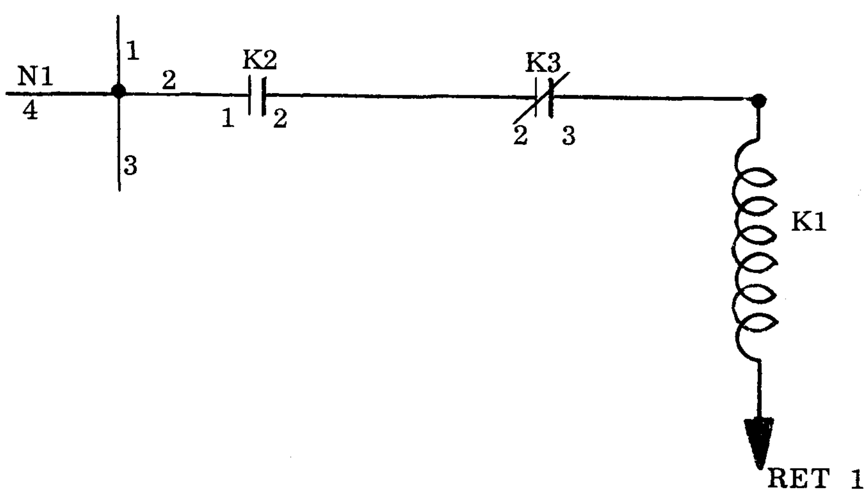


FIGURE 1-16

Describing this coil with bilateral leg equations by the format used in the last section of this report yields:

Forward: Leg (K1-N1) = Ret 1 · /CONTK3C23 * CONTK2C12 * N1 (53)

Backward: LEG2N1 = CONTK2C12 * /CONTK3C23 * Coil K1 * Ret 1 (54)

In addition, though, "Coil K1" has to be described by a Boolean equation. This is:

Coil K1 = Ret 1 * /CONTK3C23 * CONTK2C12 * N1 (55)

Note that this last equation is exactly the same as Eq. 53, the forward equation for "Leg (K1-N1)." It would be redundant and wasteful of variable usage if two variables are described by the same equation while only one of them is needed in the simulation.

3.8.3 Legs are only used in an equation to describe a node (See Section 3.3). Also, a node is only described by legs that are written in a forward direction toward the node (See Section 3.4). The forward direction of the leg named "Leg (K1 to N1)" is toward the "Coil K1." Now "Coil K1" or any other "sink" variable cannot be a node or treated as a node. So "Leg (K1 to N1)" is an unnecessary variable.

3.8.4 Every "sink" leg can be in one of two states; either normal or malfunctioned. In turn, a malfunction can be one of four types:

1. No return path supplied because of an open return circuit.
2. Return path for a power absorbing variable is at a higher potential than the source supplying this variable.
3. The "sink" leg receives an additional potential between the source of power and the power absorbing variable.
4. The "sink" leg normally malfunctions due to the opening or closing of the electrical path between the "sink" variable and its source of power.

Considering these four different malfunctioned states of a "sink" leg, for every "sink" variable in the model, two additional variables must be added. These are dummy variables called "dummy" and "NX." Every "sink" variable requires its own set of these two additional variables. The two variables added to the "sink" equations enable the equations to malfunction in the four forms listed. The completed bilateral set of leg equations for the "sink" variable of Figure 1-16 would be:

Forward

Coil K1 = Ret 1 * /CONTK3C23 * CONTK2C12 * N1 * (Nx + /Dummy) (56)

Backward

LEG2N1 = Dummy * CONTK2C12 * /CONTK3C23 * Ret 1 * (Coil K1 + Dummy) (57)

If a discrete input is a signal to a computer, a coil is generally associated with this signal. The Discrete Input must then be considered as a "sink" equation.

3.8.5 Commonly seen in parallel with coils and various other elements are diodes. When a diode is in parallel with an element, the diode should be considered an integral part of the element and therefore, not expressed as an independent variable. Conversely, when a diode is in series with other elements, as described in a previous section of this report, the diode should be considered as an independent variable.

3.9 BOOKKEEPING CLASSIFICATION OF VARIABLES

All variables can be broken down into the following classifications: (see next page)

ACTIVE — DEPENDENT	ACTIVE — INDEPENDENT	INACTIVE
Busses Nodes Grounds-Returns Legs	Sources Diodes Coils Lamps Computer Signals Dummies Legs	Terminals Permanent Connectors Grounds-Returns Busses Sources Dummies
NO/NC	NO/NC	NO/NC
Contacts Multiple Switches Automatic Connectors	Single Switches Open Semi-Permanent Connectors (Umbil-plug)	Closed Semi-Permanent Connectors (Umbil-plugs)

FIGURE 1-17

NO = Normally Open
NC = Normally Closed

Notice that some variables fall into two categories. These will be explained later in this section.

3.9.1 This information on the classification of variables will appear in the last eight columns of each time-card for each variable.

3.9.2 The Active/Inactive division was created to hold to a minimum the number of variables with which the simulation must actively work. When a large system is to be simulated with a great amount of detail, there will be more variables described than the simulation can handle at one time. But many of these variables never change state; neither in the preparations for test, nor in the test, nor in post-test. Most of these variables continuously provide a copper-path for the rest of the circuit. If this be the case, these variables are Inactive. All other variables are Active. All the variables can then be classified in this manner and a respective symbol entered into the last eight columns of a time-card for each variable. The DNS program sorts out all Inactive variables from the model so that the simulation works only with Active variables. Inactive variables are only written in the equation at first for two reasons:

1. This locates them easily in the proper leg (s) and in proper sequence so that if these variables are to be malfunctioned, it can be easily done.

2. If a pictorial display is required, all the variables can be seen.

It is interesting to note that if an Inactive variable is to be malfunctioned, it is not necessary to change the classification from Inactive to Active. The leg (s) that uses this variable can be commanded to assume a binary value of zero, and this has the same effect as that element being malfunctioned.

3.9.3 Permanent connectors are those for which the connections are not designed to be opened and closed by hand. Such connectors, when connected with tools, may be sealed to guarantee breaking and making of the connections. On the other hand, semi-permanent connectors have to be manually moved to break and make the connections. However, this can be done quite often with, say, umbilical plugs in the setting up and tearing down of particular tests.

3.9.4 Most dummies are active because they are inserted to give the model a malfunction capability, not easily describable using the variables alone. However, those dummies assigned to an unused leg are inactive. An example follows:

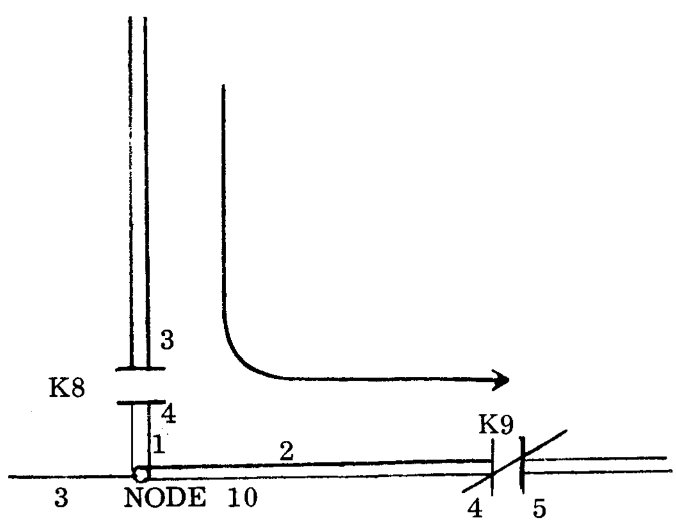


FIGURE 1-18

The double-line path and arrow indicate the copper-path actively used in the model. To show that Node 10 is really a node and not just a terminal, the unused path (LEG3N10) is named in the node equation for Node 10. Because it never changes state, it can be classified as a dummy variable in the Inactive list. Should this path ever be used, the logical node equation is correct. A logical equation for LEG3NODE 10 has to be generated and the classification of this variable must be changed.

Another deciding point for variables -- like grounds, returns, buses, and sources -- is if there is a logical equation for these variables and in this equation leg there is a active variable, then this variable in question is active; otherwise, it is inactive.

3.9.5 The Dependent/Independent division was created to supply an additional classification of all active variables. For the automatic malfunction analysis program this classification will either provide information required to do the job or check the results.

Legs are Dependent if they have dependent variables within them; otherwise, they are Independent.

3.9.6 Another division is the Normally Open/Normally Closed classification. This is only asked of such variables as contacts, switches, semi-permanent and automatic connectors or those variables that easily show their open or closed state on drawings. This classification also will aid in the automatic malfunction analysis program. In addition, this classification may be useful for future pictorial display of the system.

3.9.7 A point to consider in the classification of variables is that for a true pictorial display of the system, all variables must be classified as to what they are, not how they are written. For example, a bus many times acts like a node and if in the model a bus does so, it should be written in the format of a node equation. However, this bus should still be classified as a bus.

3.10 BOOKKEEPING CLASSIFICATION OF EQUATIONS

3.10.1 All equations can be broken down into the following classifications:

1. Bilateral Same (BS) - A leg that is logically the same in either direction of current flow. See Leg 2 Node 8, Figure 1-7, Section 3.3.2.

2. Forward Different (FD) - A leg that is logically different in both directions of current flow. This leg, however, describes the forward or normal path. See Figure 1-12 of Section 3.5.1.

3. Reverse Different (RD) - Same leg as FD but logically different and described in a reverse sequence.

4. Forward Unilateral (FU) - A leg describing an electrical copper-path in the forward direction to a power absorbing variable. See Figure 1-16 of Section 3.8.2.

5. Reverse Unilateral (RU) - Same leg as FU but logically different and described in a reverse sequence.

6. Unilateral Mechanical (UM) - A mechanical relation between two sets of contacts or a set of contacts and its coil. See Coil-Contact Relations of Section 3.6.

7. Node (X) - An equation describing a node as a function of legs connected by logical ORs. See "Node 9" of Figure 1-7, Section 3.3.2.

8. Unilateral Ground (UG) - A leg written unilaterally in the ground side of circuits. See "Ret 1" of Figure 1-15, Section 3.7.

9. Unilateral Source (US) - A leg written unilaterally from a bus or source. See "Legs 10, 11, 13, 15" of Figure 1-7, Section 3.3.2.

10. Input (I) - An input expression for a variable not normally described in the model. See "Ground" description of Section 3.7.

3.11 FORMAT FOR EQUATION-CARDS

The format for the equation cards is as follows: The variable name will occupy the columns from Col. 1 through Col. 24, with the equal sign occupying any column from Col. 7 through Col. 25. The equation will occupy any or all columns from the equal sign through Col. 72. Columns 73 through 80 will be reserved for bookkeeping information.

3.12 FORMAT FOR TIME-CARDS

The variable name will occupy columns as shown on the following page:

Variable Name (cols. 1-24)	Blank (Cols) 25-36	Pick-Up Times (cols. 37-54)			Drop-Out Times (cols. 55-72)			Bookkeeping (cols. 73-80)
		MIN	AVE	MAX	MIN	AVE	MAX	
		6 Cols.	6 Cols.	6 Cols.	6 Cols.	6 Cols.	6 Cols.	

FIGURE 1-19

3.13 BOOKKEEPING LAYOUT FOR DNS PROGRAM

3.13.1 Bookkeeping layouts used in the DNS program are as follows:

1. Col. 73 - A = Active and I = Inactive.
2. Col. 74 - D = Dependent and I = Independent.
3. Cols. 75 & 76 = Type of equation, as determined from the classification guide on Page 28 of this report.
4. Col. 77 = Not used.
5. Col. 78 = P used when coil has diode in parallel.
6. Col. 79 = Type variable, as coded from Pages 32 and 33 of this report.
7. Col. 80 = 0 if element is normally open or de-energized and 1 if element is normally closed or energized.

8. The page numbers of the drawing from which the equation started are shown in columns 73, 74, 75, and 76 of the equation card.

9. The sequence numbers of the equation written from the drawing shown in columns 73, 74, 75 are shown in columns 77 and 78 of the equation card.

10. The cards required for equations are numbered in sequence and are shown in columns 79 and 80 of the equation cards.

Code Designation For Cols. 73-80 Of Time-Cards	Active Inactive 73	Dependent Independent 74	Type of Equation 75	76	77	Diode Parallel 78	Type Variation 79	N/C N/O 80
Accelerometer				I			A	
Bus	A	I		I			B	0
Contact	A	D	U	M			C	1/0
Diode	A	I		I			D	1/0
Discrete In	A	I	F	U			I	0
Coil	A	I	F	U		P	K	0
Light	A	I	F	U			L	0
Solenoid	A	I	F	U			N	0
Discrete Out	A	I		I			O	0
Pin	I						P	1
Switch	A	D/1		I			S	0
Timer	A	I	F	U			T	0
Dummy	A	I		I			U	0
Node	A	D		X			X	0
Leg	A						Y	0
Sensor	A			I			Z	0
Input	A			I			0	0

	Bilateral, Same	B	S
	Forward Different	F	D
	Reverse Different	R	D
Leg Designation as Shown on Pages 28 & 29.	Forward Unilateral	F	U
	Reverse Unilateral	R	U
	Mechanical Unilateral	U	A
	Ground Unilateral	U	G
	Source Unilateral	U	S
	Input		I

TABLE 2

Code Designation For Column 79 Of Time-Cards

	Col. 79
Accelerometer	A
Bus	B
Contact	C
Diode	D
	E
Flt. Comb. Monitor	F
	G
Heater	H
Discrete In	I
Transducer	J
Relay Coil	K
Light	L
Meter	M
Solenoid	N
Discrete Out	O
Pin	P
	Q
	R
Switch	S
Timer	T
Dummy	U
Valve	V
	W
Node	X
Leg	Y
Sensor	Z
	1
	2
	3
	4
	5
	6
	7
	8
	9
	0

TABLE 3

3.14 CONSISTENCY IN BUILDING THE MODEL

Consistency is of greatest importance in building a model. Variables must be designated by the exact letters and digits in each place in which they appear, both in equation and time parameters. Legs of nodes should be numbered in the same manner by all persons writing a particular model, so that interface equations are written the same.